CNG 514
Assignment 1: Understanding Data and Preprocessing

Student Name: Furkan Duman
Student Number: 2453173
Date of Submission: 23 March 2024

Contents

1 Identify and characterize a dataset **4**

2 Identify and characterize attributes **12**
2.1 Boxplots 18
2.2 Histogram 21
2.3 Scatter Plot 26

3 Data preprocessing **27**
3.1 Binning and Smoothing Part 27
3.2 Normalization 31
3.3 Correlation 34

4 References **36**

List of Figures

1 Calculation of the Central Tendency and Dispersion Measures . . 12
2 Boxplot for AnnualHouseholdIncome before cleaning 13
3 CoronavirusConcern2 14
4 Detect empty values 14
5 Count of the missing values 15
6 Delete the missing values 15
7 Trimming Operation 15
8 AnnualHouseholdIncome Measures 16
9 BirthYear2020 Measures 16
10 Age Measures 17
11 CoronavirusConcern2 Measures 17
12 Boxplot function 18
13 Boxplot for AnnualHouseholdIncome after cleaning 18
14 Boxplot for BirthYear2020 after cleaning 19
15 Boxplot for Age after cleaning 20
16 Boxplot for CoronavirusConcern2 after cleaning 21
17 Histogram Function 21
18 Histogram for AnnualHouseholdIncome after cleaning 22
19 Histogram for BirthYear2020 after cleaning 23
20 Histogram for Age after cleaning 24
21 Histogram for CoronavirusConcern2 after cleaning 25
22 Scatter function 26
23 Scatter Plot for Age vs AnnualHouseholdIncome 26
24 Binning 27
25 Binning CoronavirusConcern2 27
26 Equal-width Binning for Annual Household Income 28
27 Histogram of Coronavirus Concern Level With Binning 28
28 Code-Discretize Age 29
29 Histogram of Discretize Age 30
30 Function of Max-Min normalization and Z-score normalization . 31
31 Histogram Of Normalized AnnualIncome 32
32 Histogram Of Normalized Age 32
33 Histogram Of Normalized AnnualIncome(With Z Score) 33
34 Histogram Of Normalized Age(With Z Score) 33
35 spearman$_c orrelation_w ith_p$ 34
36 pearson$_c orrelation_w ith_p$ 34
37 Calling Correlation Functions 35
38 Correlation Results 35

Abstract

This report presents the analysis and preprocessing of a dataset containing COVID-19 survey data collected in the US. The dataset consists of over 10,000 participant records with various attributes related to demographics, concerns, intentions, and beliefs regarding the pandemic. The main objectives of this assignment are to identify and characterize the dataset, perform data preprocessing techniques, and explore visualization methods to gain insights into the data.

Additionally, normalization techniques are applied to the dataset to ensure consistency and comparability among attributes. Both min-max normalization and z-score normalization methods are implemented to scale the values of selected attributes to a common range.

Furthermore, the correlation between attributes is calculated to understand the relationships between them. Specifically, the correlation between AnnualHouseholdIncome and selected attributes is examined to identify any potential associations. This analysis provides valuable insights into how different variables may be related and helps in identifying patterns within the dataset.

1 Identify and characterize a dataset

The dataset provided by us, named cng514-covid-survey-data.csv, includes information from more than 10,000 data objects. This dataset contains information about people and their actions during the COVID-19 period encapsulating a total of 20 attributes. It includes details like personal characteristics, behaviors, and the precautions individuals took to prevent the spread of the disease. The data provides insights into how different groups of people responded to the pandemic, making it a valuable resource for understanding health-related patterns during this time.

It is organized as follows for each participant and includes these attributes:

1. **ParticipantId**
   - This is an attribute used to identify each participle.
   - It is a nominal attribute. The first participant has been assigned the value of 0, while the last participant has a value of 10753.
2. **AnnualHouseholdIncome**
   - For each participant, this attribute show the annual household income.
   - It is a ratio attribute.
   - There are some anomalies in this attribute. For example, some values are less than 0, and some values are empty. This data must be cleaned before data mining.
3. **BirthYear2020**
   - This records the birthdate of each participant using a four-digit number.
   - It is a Interval attribute. The range spans from 0 to 2002.
   - There are some anomalies in this attribute. For example, some values are 0, some values are empty. This data must be cleaned before data mining.

4. **CoronavirusConcern2**

   - This records the birthdate of each participant using a four-digit number.
   - It is an ordinal attribute, as the values represent a meaningful order from 'Not at all concerned' (0) to 'Extremely concerned' (10).
   - There are some anomalies in this attribute. For example, some values are not in range (-0.5) , some values are decimal, some values are empty. This data must be cleaned before data mining.

5. **CoronavirusEmployment**

   - This attribute illustrates the changes in participants' employment statuses since 2020.
   - It is an nominal attribute. It should be:

   • was-full: I was employed full-time on January 1, 2020.
   • was-part: I was employed part-time on January 1, 2020
   • was-jobless: I was unemployed on January 1, 2020.
   • now-full: I am now employed full-time
   • now-part: I am now employed part-time.
   • now-jobless: I am now unemployed.
   • now-retired: I am now retired.
   • was-retired: I was retired on January 1, 2020
   • was-disabled: I was disabled and unable to work on January 1, 2020.
   • now-disabled: I am now disabled and unable to work.
   • There are some anomalies in this attribute. For example, some values are empty. This data must be cleaned before data mining.

6. **CoronavirusIntent_Mask**

   - This attribute indicates the participant's intention to wear a mask in public.
   - It is an ordinal attribute, as the values represent a meaningful order from 0 (No) to 100 (Yes).
   - There are some anomalies in this attribute. For example, some values are not in range (-5) , some values are empty. This data must be cleaned before data mining.

7. **CoronavirusIntent_SixFeet**

   - This attribute indicates the participant's intention to maintain a six-feet distance from other people.
   - It is an ordinal attribute, as the values represent a meaningful order from 0 (No) to 100 (Yes).
   - There are some anomalies in this attribute. For example, some values are not in range (-5, 250, 300) , some values are empty. This data must be cleaned before data mining.

8. **CoronavirusIntent_StayHome**

   - This attribute indicates the participant's intention to stay at home as much as possible.
   - It is an ordinal attribute, as the values represent a meaningful order from 0 (No) to 100 (Yes).
   - There are some anomalies in this attribute. For example, some values are not in range (-10, 250, 300) , some values are empty. This data must be cleaned before data mining.

9. **CoronavirusIntent_WashHands**

   - This attribute indicates whether the participant intends to wash their hands more than usual, for at least 20 seconds each time.
   - It is an ordinal attribute, as the values represent a meaningful order from 0 (No) to 100 (Yes).
   - There are some anomalies in this attribute. For example, some values are not in range (-10, 250, 300) , some values are empty. This data must be cleaned before data mining.

10. **CoronavirusLocalCommunity**

    - This attribute indicates whether the participant knows anyone in their local community who has contracted Coronavirus.
    - It is an ordinal attribute; if applicable, it records the number of people, with '0' indicating none.
    - There are some anomalies in this attribute. For example, some values are empty. This data must be cleaned before data mining.

11. CoronavirusSupportSystem

- This attribute indicates whether the participant believes they will need support in the next 6 months due to the pandemic. If so, it prompts them to identify the person or group they consider most likely to help them.
- It is an nominal attribute. It can be:

- fam-friend: Family and friends
- employer: Employer
- religious: Religious community
- local-gov: Local government
- state-gov: State government
- fedgov: Federal government
- other: Other
- no-one: No one
- local-community: Local community groups
- private-org: A private organization
- There are some anomalies in this attribute. For example, some values are empty. This data must be cleaned before data mining.

12. CoronavirusSymptomSelect

- This attribute indicates whether the participant personally experiences any symptoms.
- It is an nominal attribute. It can be:

- dry-cough: Dry cough
- short-breath: Shortness of breath
- diarrhea: Diarrhea
- muscle-ache: Muscle ache
- fatigue: Fatigue
- nasal: Runny nose or nasal congestion
- sore-throat: Sore throat
- lost-smell-taste: Loss of smell / taste
- fever: Fever
- headache: Headache
- nausea-vomit: Nausea and/or vomiting

- none: None
- There are some anomalies in this attribute. For example, some values are empty. This data must be cleaned before data mining.

13. Education_Alt2

– This attribute illustrates the educational background of the participant.

– It is an ordinal attribute. It can be:

- school: Some School / No Diploma
- highschool: High School Graduate
- some-college: Some College
- college: College Degree
- postgrad: Postgraduate Degree
- There are some anomalies in this attribute. For example, some values are not in range (1, 6). Some values are empty. This data must be cleaned before data mining.

14. Ethnicity

– This attribute indicates the race or ethnic group of the participant.

– It is an nominal attribute. It can be:

- asian: Asian
- black: Black
- hispanic-latino: Hispanic or Latino
- white: White
- other-mixed: Other/Mixed
- There are some anomalies in this attribute. For example, some values are not in range (1, 9). Some values are empty. This data must be cleaned before data mining.

15. Gender

- This attribute displays the gender of the participant
- It is an nominal attribute. It can be:

- female
- male
- other
- There are some anomalies in this attribute. For example, some values are not in range (1, 2). Some values are empty. This data must be cleaned before data mining.

16. HasCoronavirusBelief

- This attribute indicates whether the participant believes they have coronavirus or not.
- It is an ordinal attribute, as the values represent a meaningful order from 0 (Definetely No) to 100 (Yes).
- There are some anomalies in this attribute. For example, some values are written as a decimal, some values are not, and same values are empty. This data must be cleaned before data mining.

17. HasCoronavirusBelief

- This attribute represents the political beliefs of the participant.
- It is an ordinal attribute, as the values represent a meaningful order from 0 (Very Liberal) to 10 (Very Conservative).
- There are some anomalies in this attribute. For example, some values are written as a decimal, some values are not, and some values are empty. This data must be cleaned before data mining.

18. Religion_Alt1

- This attribute displays the participant's religion, if any.
- It is an nominal attribute. It can be:

- evangelical-protestant: Evangelical Protestant
- other-protestant: Other Protestant

- catholic: Catholic
- mormon: Mormon
- orthodox: Orthodox
- jewish: Jewish
- muslim: Muslim
- buddhist: Buddhist
- hindu: Hindu
- atheist: Atheist
- agnostic: Agnostic
- something-else: Something Else
- nothing-in-particular: Nothing in Particular
- There are some anomalies in this attribute. For example, some values are not in range (1). Some values are empty. This data must be cleaned before data mining.

19. Religiosity_Alt2

- This attribute reflects the participant's perception of the importance of religion in their life.
- It is an ordinal attribute, as the values represent a meaningful order from 0 (Not Very Important) to 10 (very important).
- There are some anomalies in this attribute. For example, some values are written as a decimal, some values are not, and some values are empty, some values are not in range (-5) . This data must be cleaned before data mining.

20. ZipCode

- This attribute indicates the participant's ZIP code.
- It is an nominal attribute.
- There are some anomalies in this attribute. For example, some values are not, and some values are empty, some values are not in range (-200, -187) . This data must be cleaned before data mining.

Multiple methods can be used when analyzing this data set. Data mining methods such as **classification** and **clustering** allow us to obtain various information by providing different perspectives.[[1]] Using a classification method when analyzing this data set can provide valuable information to predict participants' behavior in compliance with COVID-19 precautions. **Classification** is used for the purpose of predicting a specific target variable. For example, it is possible to divide into two classes such as "Behaving Appropriately" or "Not Behaving Appropriately". When using this method, we can use attributes such as CoronavirusIntent_Mask, CoronavirusIntent_SixFeet, CoronavirusIntent_StayHome, CoronavirusIntent_WashHands. As a potential result in this analysis, we can learn the tendency of each user to comply with COVID-19 precautions, and we can learn to what extent the use of masks or other attributes is important in which age range or demographic groups. Using this data, the public can be guided to take precautions against COVID-19.

2 Identify and characterize attributes

In this section, I have selected the attributes **AnnualHouseholdIncome** (Ratio Data Type), **BirthYear2020** (Interval Data Type), and **Coronavirus-Concern2** (Ordinal Data Type) for performing computations. To compute the central tendency and dispersion values, I utilized the Pandas module in Python. To create boxplots, histograms, and scatter plots for measuring the dispersion of data, I used the Matplotlib library.

- **Central Tendency and Dispersion Measures**

  I developed a function to calculate measures of central tendency and dispersion measures for a given column in a data set. The function takes the specified column and its name, and **its attribute type** as input and calculates basic statistical metrics such as mean, median, mode, range, quartiles, and variance. Then, it displays these values.

```
def find_CentralTendency_AND_DispersionMeasures(column,column_name,column_data_type):

    if column_data_type == "nominal":
        mode_value = column.mode().iloc[0]

    elif column_data_type == "ordinal":
        mode_value = column.mode().iloc[0]
        median_value = column.median()
        quartiles = column.quantile([0.25, 0.50, 0.75]).to_dict()
        variance_value = column.var()


    elif column_data_type == "interval" or column_data_type == "ratio":
        mode_value = column.mode().iloc[0]
        median_value = column.median()
        mean_value = column.mean()
        max_value = column.max()
        min_value = column.min()
        range_value = max_value - min_value
        quartiles = column.quantile([0.25, 0.50, 0.75]).to_dict()
        variance_value = column.var()
        CreateBoxPlot(column=column,column_name=column_name)
        CreateHistogram(column=column,column_name=column_name)
```

Figure 1: Calculation of the Central Tendency and Dispersion Measures

As seen in this figure, I have designed a function that calculates specific values for each attribute type(Nominal-Ordinal-Interval-Ratio).

However, **I need to clean the data before calling this function**. This is because outliers or null values can negatively affect the calculations. When I look at the boxplot, we can observe that the attributes have outlier data points.

– **AnnualHouseholdIncome**

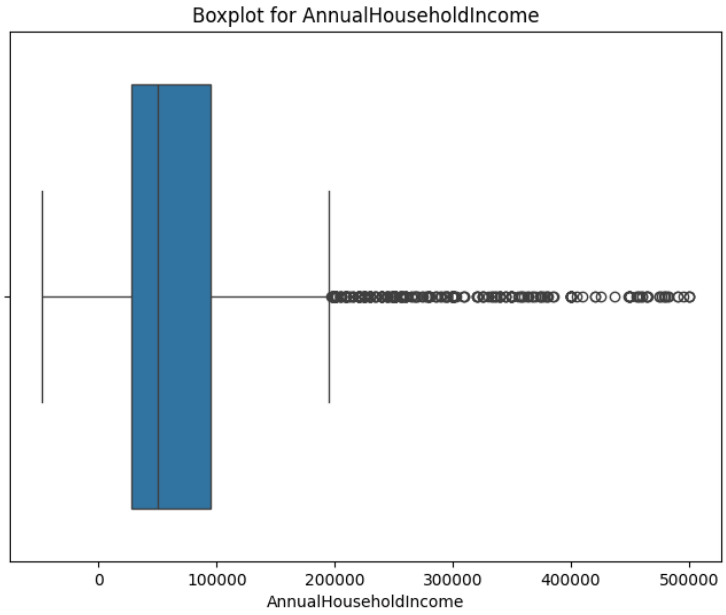

Figure 2: Boxplot for AnnualHouseholdIncome before cleaning

– **CoronavirusConcern2**

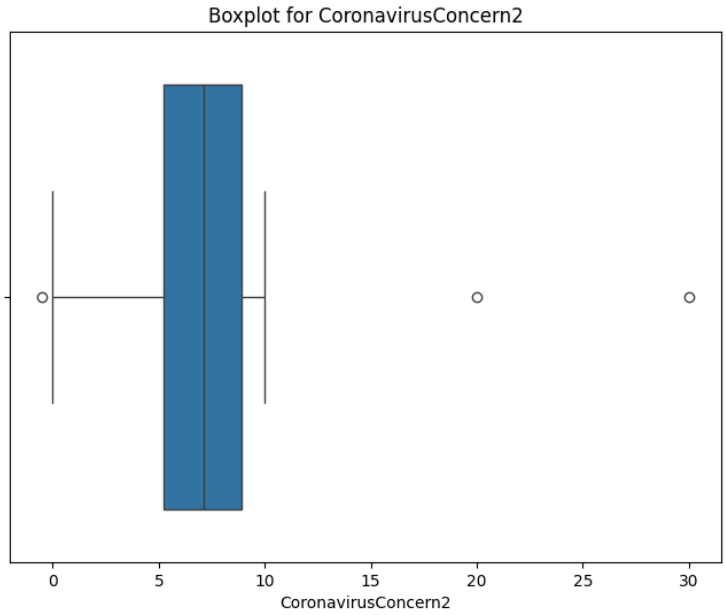

Figure 3: CoronavirusConcern2

In these figures, we can easily see that there are outlier values in the selected attributes. So we need the clean the data.

- **Empty Values**

To clean the data, I first checked if there were **any missing values** in the attribute data.

```
3 usages
def calculate_Empty_values(data,columnName):
    return data[columnName].isnull().sum()
```

Figure 4: Detect empty values

In this figure, I designed a function to count the number of empty values within the attributes.

```
Missing Values:
AnnualHouseholdIncome: 2, BirthYear2020: 2, CoronavirusConcern2: 3
```

Figure 5: Count of the missing values

As you can see in the figure, there are a few empty values. Since the number of these missing values is small, I removed them from the dataset.

```
After delete missing values:
AnnualHouseholdIncome: 0, BirthYear2020:  0, CoronavirusConcern2: 0
```

Figure 6: Delete the missing values

In this figure, we can see that, there are no empty value anymore.

After that, I decided to perform a '**trimming**' operation. I use the following function to perform the trimming operation.

```
def trimmingOperation(data, lower_percentile=5, upper_percentile=95):

    lower_index = int(lower_percentile / 100 * len(data))
    upper_index = int(upper_percentile / 100 * len(data))

    trimmed_data = data[lower_index:upper_index]

    return trimmed_data
```

Figure 7: Trimming Operation

In this figure, we can see trimmingOperation function. I designed this function to perform a trimming operation on a dataset. It takes in a dataset as input along with optional parameters for the lower and upper percentiles, which default to 5% and 95% respectively. The function calculates the indices corresponding to these percentiles based on the length of the dataset. Then, it trims the dataset by selecting the data points between the calculated indices. Finally, the trimmed dataset is returned.

Now, **attributes are cleaned**. Now, I can calculate the central tendency and dispersion measures!

```
column_AnnualHouseholdIncome Measures:
************************************************************
Mode: 50000.0
Median: 50000.0
Mean: 64401.98233440115
Max value:200000.0, Min value: 10000.0, and  Range is: 190000.0
Variance: 1867910642.480082
Quartiles:
 {0.25: 30000.0, 0.5: 50000.0, 0.75: 90000.0}
************************************************************
```

Figure 8: AnnualHouseholdIncome Measures

```
column_BirthYear2020 Measures:
************************************************************
Mode: 2000.0
Median: 1985.0
Mean: 1983.385036684923
Max value:2001.0, Min value: 1955.0, and  Range is: 46.0
Variance: 158.50200013559083
Quartiles:
 {0.25: 1974.0, 0.5: 1985.0, 0.75: 1995.0}
************************************************************
```

Figure 9: BirthYear2020 Measures

```
Age Measures:
**********************************************************
Mode: 20.0
Median: 35.0
Mean: 36.614963315076984
Max value:65.0, Min value: 19.0, and  Range is: 46.0
Variance: 158.50200013559083
Quartiles:
 {0.25: 25.0, 0.5: 35.0, 0.75: 46.0}
**********************************************************
```

Figure 10: Age Measures

```
column_CoronavirusConcern2 Measures:
************************************************************
Median: 7.1
Mode: 10.0
Quartiles:
 {0.25: 5.2, 0.5: 7.1, 0.75: 8.9}
Variance: 7.106292783120269
************************************************************
```

Figure 11: CoronavirusConcern2 Measures

2.1 Boxplots

I wrote this function for computing boxplots:

```
def CreateBoxPlot(column,column_name):

    plt.figure(figsize=(8, 6))
    sns.boxplot(x=column)
    plt.title(f'Boxplot for {column_name}')
    plt.show()
```

Figure 12: Boxplot function

Boxplots:

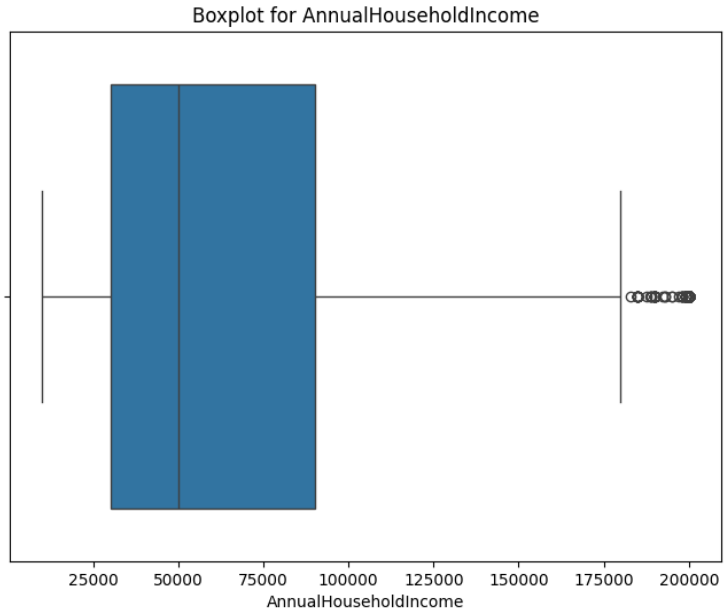

Figure 13: Boxplot for AnnualHouseholdIncome after cleaning

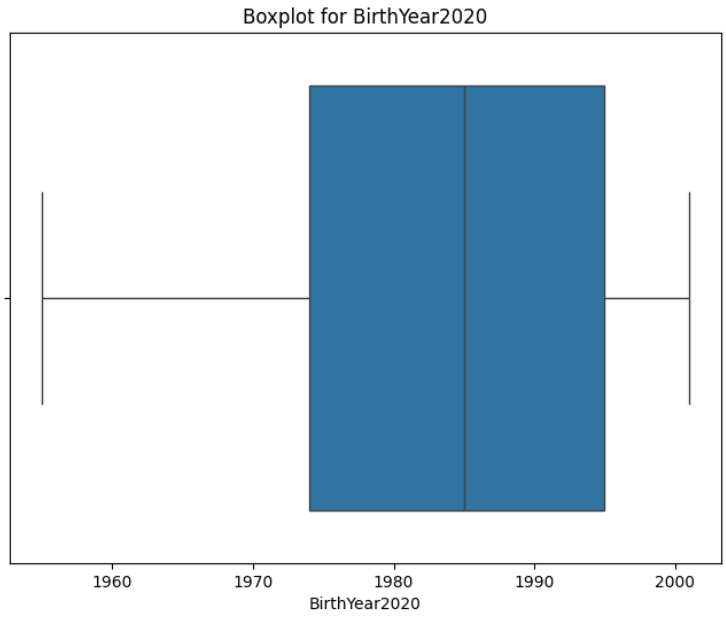

Figure 14: Boxplot for BirthYear2020 after cleaning

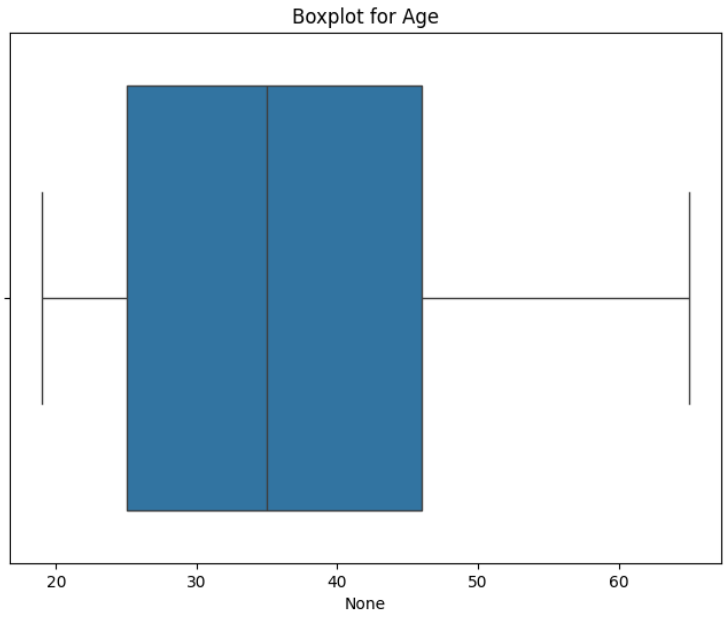

Figure 15: Boxplot for Age after cleaning

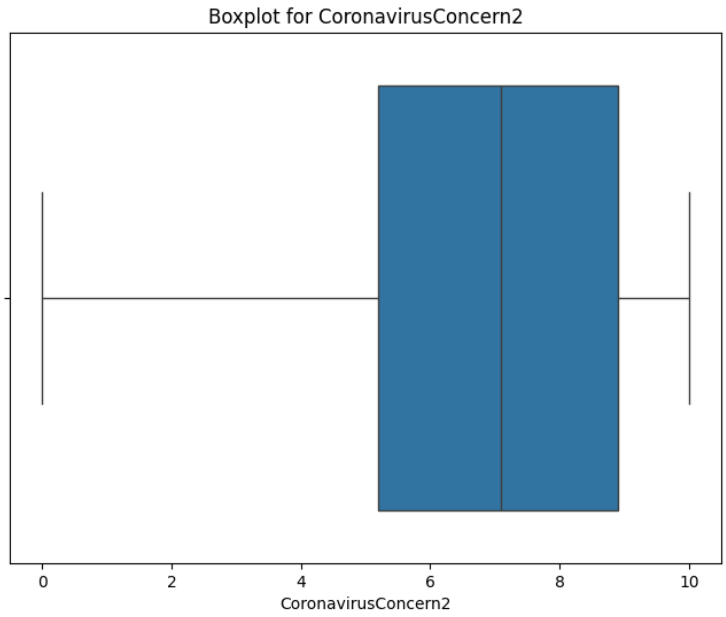

Figure 16: Boxplot for CoronavirusConcern2 after cleaning

2.2 Histogram

I wrote this function for computing Histogram:

```
def CreateHistogram(column,column_name):

    plt.figure(figsize=(8, 6))
    plt.hist(column, bins=20, edgecolor='black')
    plt.title(f'Histogram for {column_name}')
    plt.xlabel(f'{column} Values')
    plt.ylabel('Frequency')
    plt.show()
```

Figure 17: Histogram Function

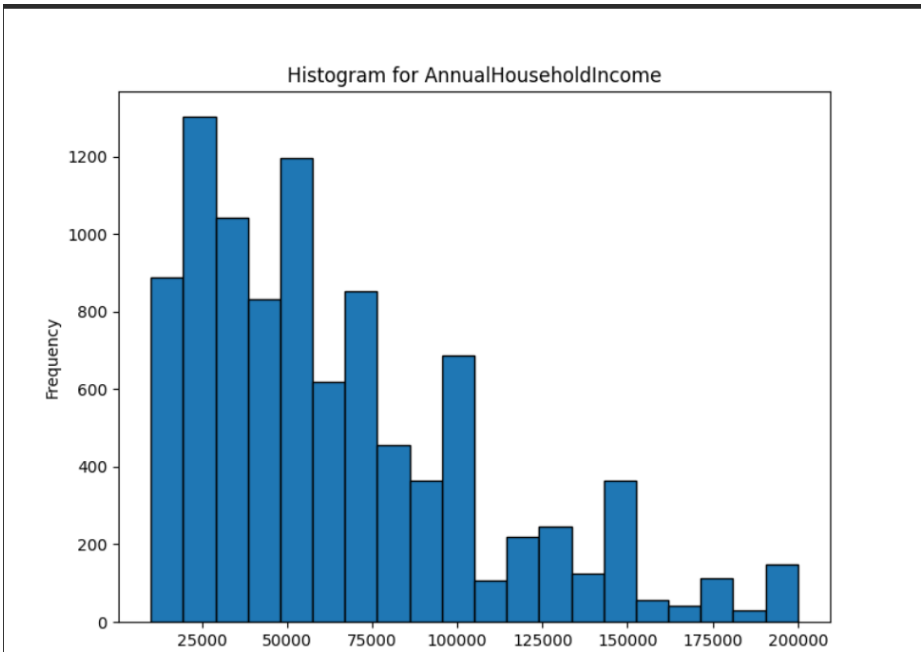

Figure 18: Histogram for AnnualHouseholdIncome after cleaning

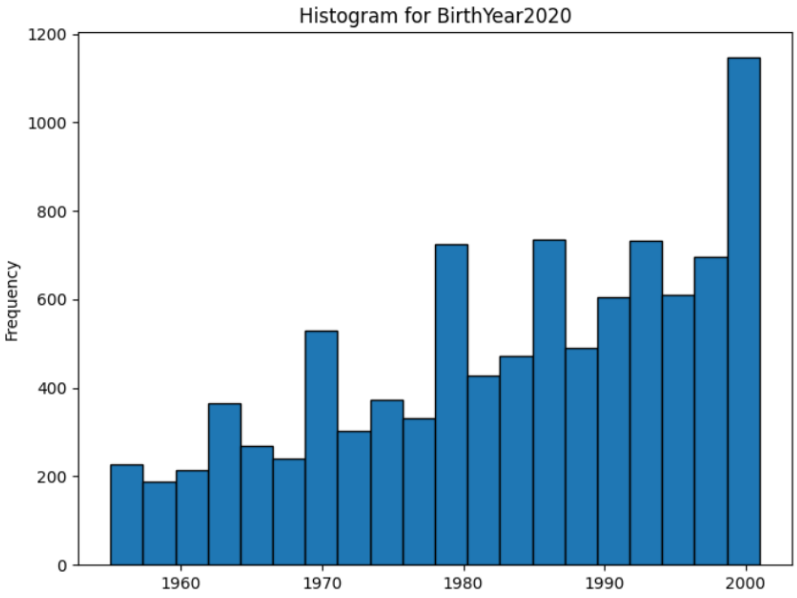

Figure 19: Histogram for BirthYear2020 after cleaning

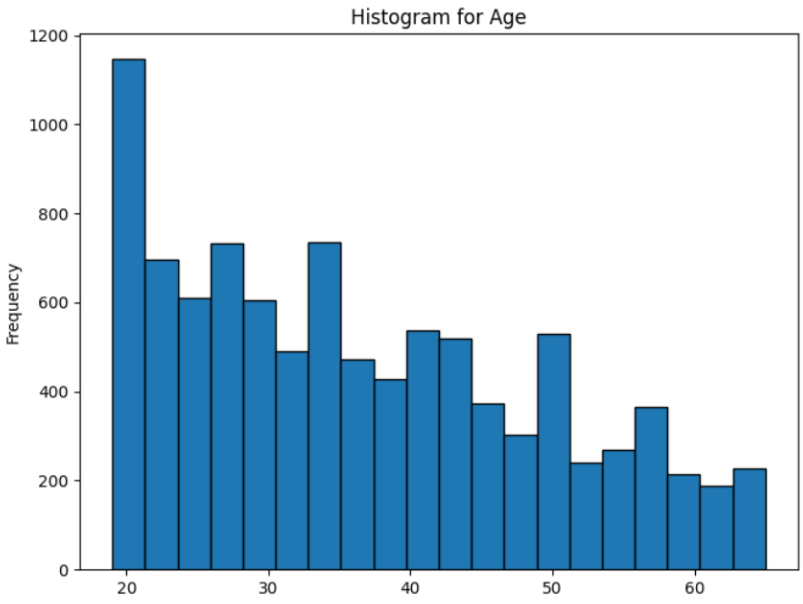

Figure 20: Histogram for Age after cleaning

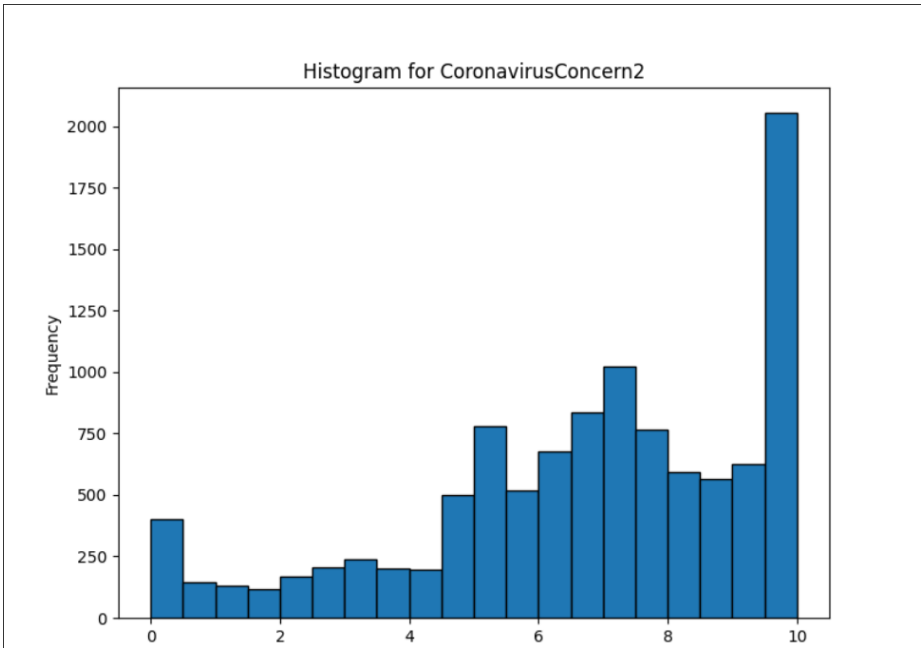

Figure 21: Histogram for CoronavirusConcern2 after cleaning

2.3 Scatter Plot

I wrote this function for computing Histogram:

```
def CreateScatterPlot(x_column, y_column):
    plt.figure(figsize=(8, 6))
    plt.scatter(data[x_column], data[y_column])
    plt.title(f'Scatter Plot for {x_column} vs {y_column}')
    plt.xlabel(f'{x_column} Values')
    plt.ylabel(f'{y_column} Values')
    plt.show()
```

Figure 22: Scatter function

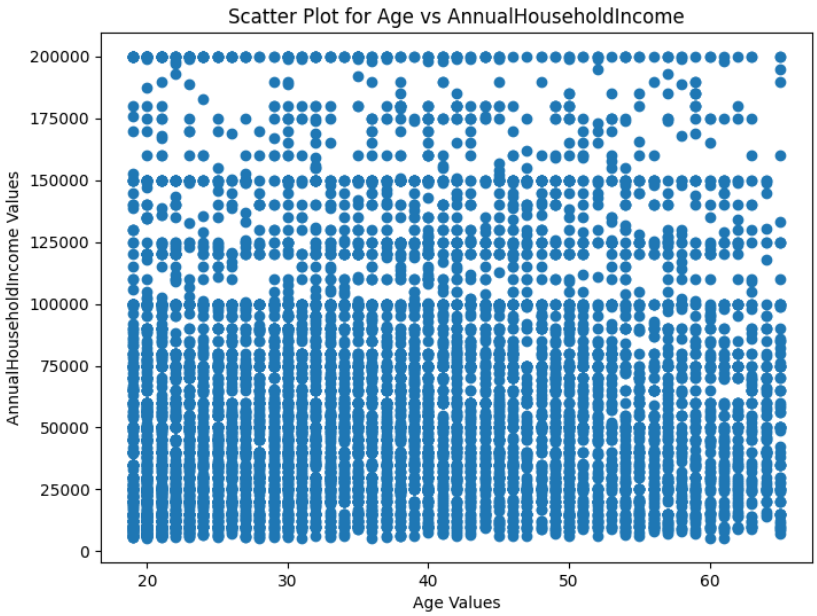

Figure 23: Scatter Plot for Age vs AnnualHouseholdIncome

3 Data preprocessing

In part 1, I talked about the anomilies I see in every data type. In the second part, anomilies were quite evident, especially in the boxplot and histogram graphics, so I cleaned the attributes. Now, for smooting data, I can use **binnig method**.

3.1 Binning and Smoothing Part

Binning is a data preprocessing technique used to smooth noisy data by grouping continuous values into a smaller number of intervals, or bins, and replacing the values within each bin with a representative value.

```
num_bins = 10
binned_data, bin_edges = numpy.histogram(processed_annualincome, bins=num_bins)
binned_Annunal = numpy.repeat(bin_edges[:-1], binned_data)


binned_data, bin_edges = numpy.histogram(processed_CoronavirusConcern2, bins=num_bins)
binned_CoronavirusConcern2 = numpy.repeat(bin_edges[:-1], binned_data)

binned_data, bin_edges = numpy.histogram(processed_BirhYear2020, bins=num_bins)
binned_BirhYear2020 = numpy.repeat(bin_edges[:-1], binned_data)
```

Figure 24: Binning

I designed a binning process to discretize three different attributes in the dataset: processed_annualincome, processed_CoronavirusConcern2, and processed_BirhYear2020.

```
bins = [0, 3, 6, numpy.inf]
labels = ['Not at all', 'Somewhat', 'Extremely concerned']

hist, _ = numpy.histogram(processed_CoronavirusConcern2, bins=bins)
```

Figure 25: Binning CoronavirusConcern2

I also designed a binning process for CoronavirusConcern2.

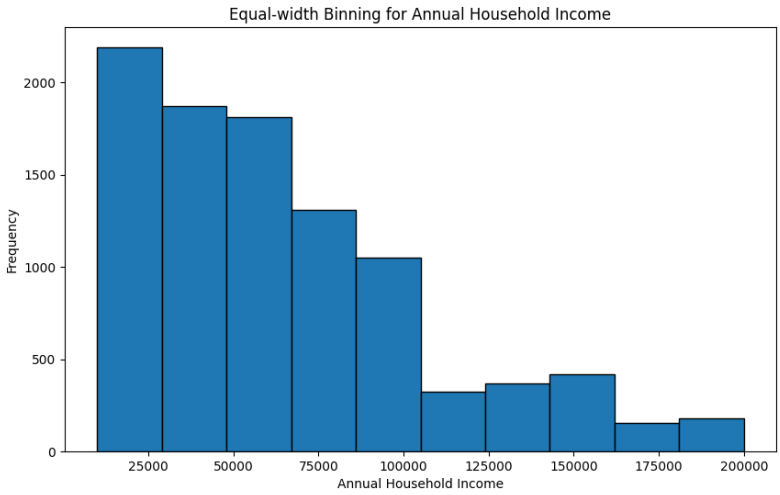

Figure 26: Equal-width Binning for Annual Household Income

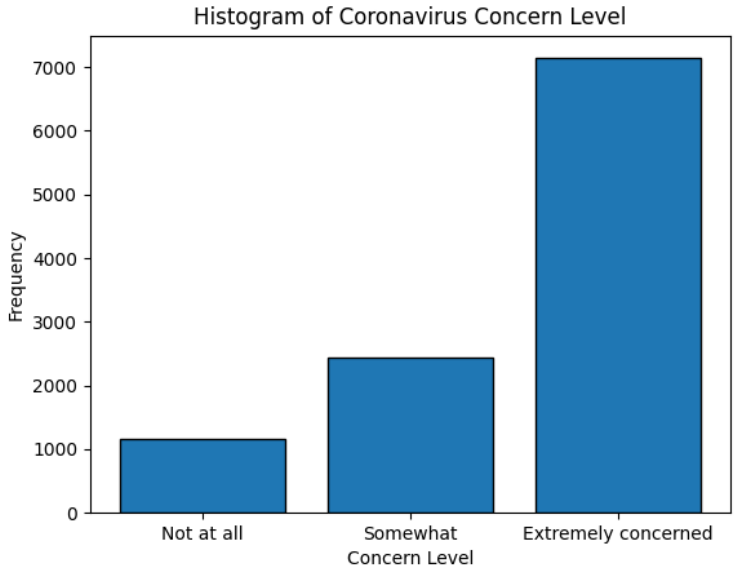

Figure 27: Histogram of Coronavirus Concern Level With Binning

```
currentAge = [2020 - int(i) for i in processed_BirhYear2020]

age_bins = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
age_labels = [f"{age}-{age+10}" for age in age_bins[:-1]]
age_labels.append(f"{age_bins[-2]}+")


plt.hist(currentAge, bins=age_bins, edgecolor='black')
plt.xlabel('Age Range')
plt.ylabel('Number of Individuals')
plt.title('Age Distribution')

plt.xticks(age_bins, age_labels)
plt.grid(True)
plt.show()
```

Figure 28: Code-Discretize Age

I also discretize age.

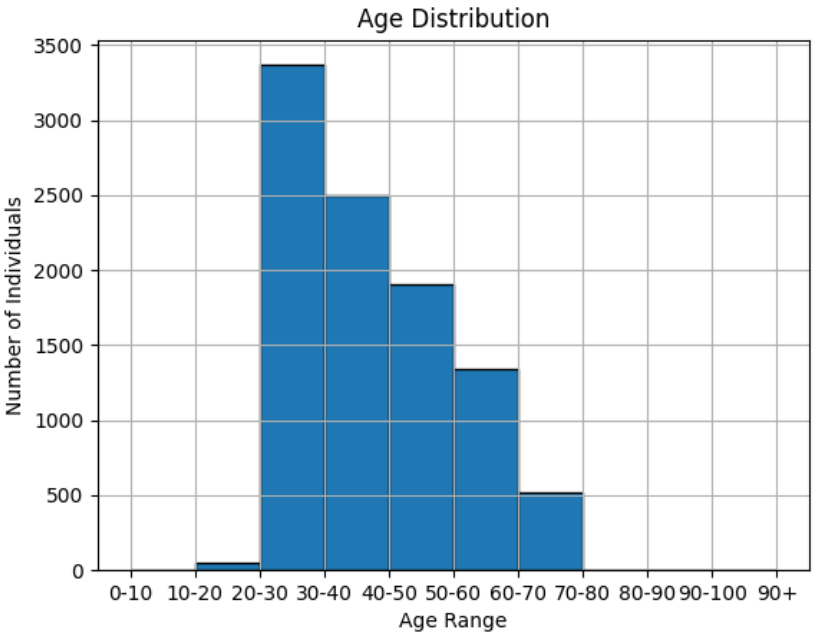

Figure 29: Histogram of Discretize Age

3.2 Normalization

To standardize the attributes and ensure uniformity in their scales, I applied two common normalization techniques: min-max normalization and z-score normalization.

Normalization is applied to rescale the ranges of numerical data, and it is generally not applied to categorical data. **Therefore, "CoronavirusConcern2" was not subjected to normalization**. However, "AnnualHouseholdIncome" and "Age" data were normalized as they are suitable for normalization.

```
def max_min_normalization(series):

    min_val = series.min()
    max_val = series.max()
    normalized_series = (series - min_val) / (max_val - min_val)
    return normalized_series


def z_score_normalization(series):

    z_scores = stats.zscore(series)
    normalized_series = pd.Series(z_scores, index=series.index)
    return normalized_series
```

Figure 30: Function of Max-Min normalization and Z-score normalization

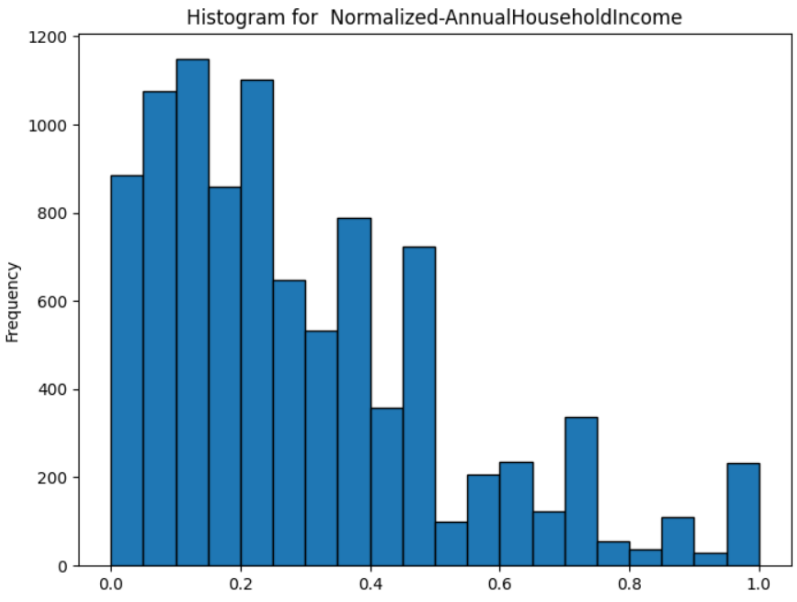

Figure 31: Histogram Of Normalized AnnualIncome

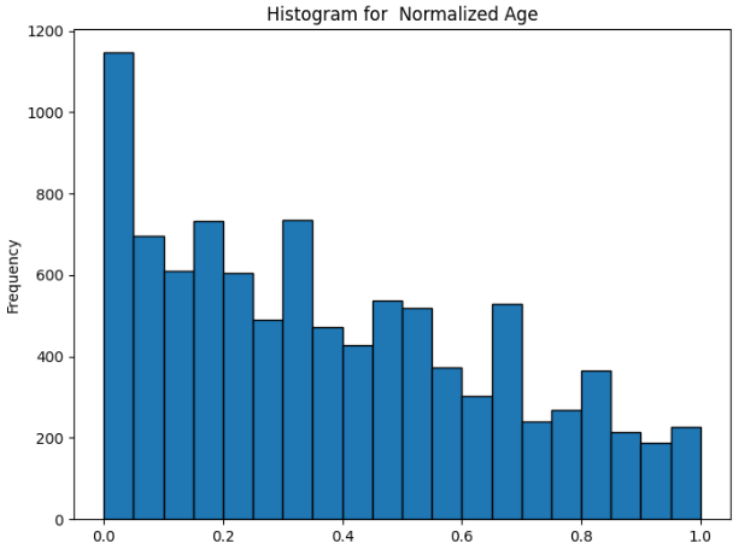

Figure 32: Histogram Of Normalized Age

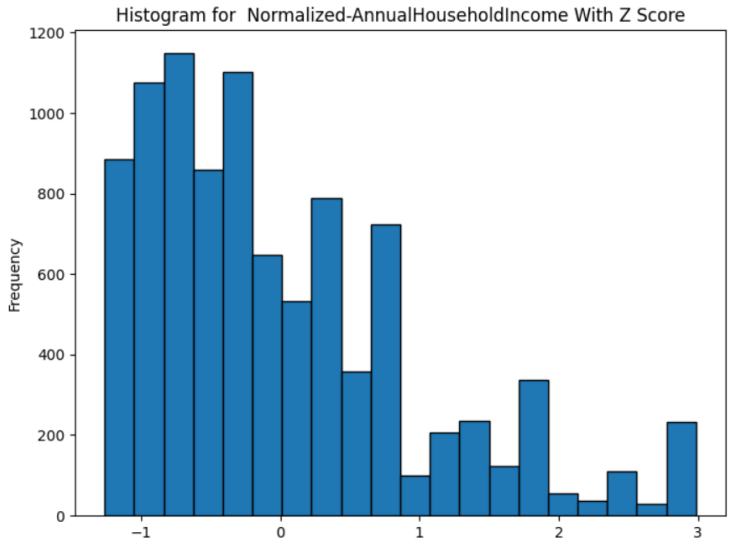

Figure 33: Histogram Of Normalized AnnualIncome(With Z Score)

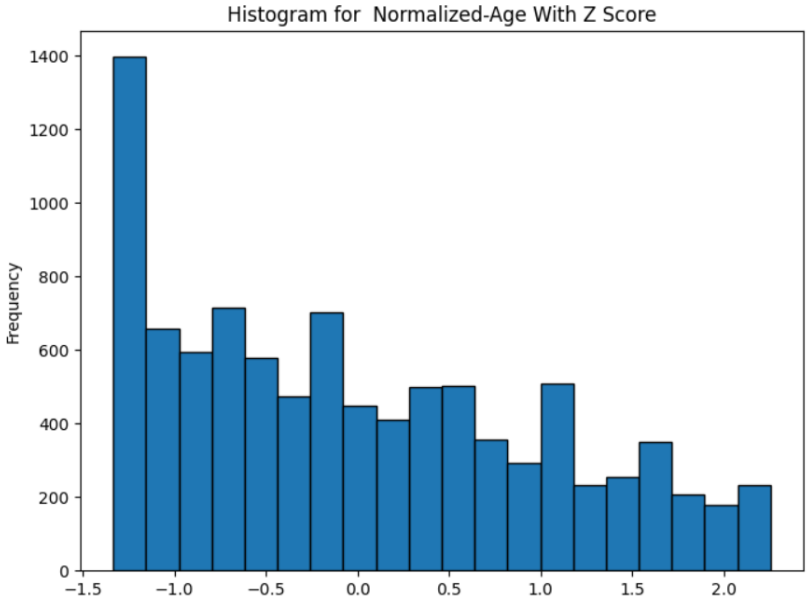

Figure 34: Histogram Of Normalized Age(With Z Score)

3.3 Correlation

For the correlation between 'AnnualHouseholdIncome' (continuous) and 'Age' (continuous), I chose to apply Pearson correlation. This is suitable because Pearson correlation measures the linear relationship between two continuous variables. In this case, it assesses how the change in one variable (e.g., age) relates to the change in another variable (e.g., income), assuming a linear relationship.

For the correlation between 'AnnualHouseholdIncome' (continuous) and 'CoronavirusConcern2' (ordinal), I chose to apply Spearman correlation. This is appropriate because Spearman correlation assesses the monotonic relationship between two variables. In our case, 'CoronavirusConcern2' is an ordinal variable, meaning it has a meaningful order but the intervals between the categories may not be uniform. Using Pearson correlation accounts for this ordinal nature and assesses the overall monotonic trend between the income and concern levels without assuming a linear relationship.

```
def spearman_correlation_with_p(x, y):

    spearman_corr, p_value = spearmanr(x, y)
    return spearman_corr, p_value
```

Figure 35: spearman$_c orrelation_w ith_p$

```
def pearson_correlation_with_p(x, y):

    pearson_corr, p_value = pearsonr(x, y)
    return pearson_corr, p_value
```

Figure 36: pearson$_c orrelation_w ith_p$

```
correlation_3, p_value_3 = spearman_correlation_with_p(normalized_AnnualIncome_series, normalized_CoronaConcern2)
print("Spearman Correlation Coefficient:", correlation_3)
print("P Value:", p_value_3)


correlation_pearson, p_value_pearson = pearson_correlation_with_p(normalized_AnnualIncome_series, normalized_age_series)
print("Pearson Correlation Coefficient:", correlation_pearson)
print("P Value:", p_value_pearson)
```

Figure 37: Calling Correlation Functions

```
Spearman Correlation Coefficient: -0.1316101429171041
P Value: 2.735543322435156e-38
Pearson Correlation Coefficient: 0.0989697297001428
P Value: 2.706462791003944e-22
```

Figure 38: Correlation Results

The Spearman correlation coefficient of -0.1316101429171041 suggests a weak negative monotonic relationship between the variables, while the Pearson correlation coefficient of 0.0989697297001428 indicates a weak positive linear relationship. Both p-values, 2.735543322435156e-38 and 2.706462791003944e-22 respectively, are extremely small, indicating that these relationships are statistically significant and unlikely to have occurred by random chance alone. Therefore, despite the weak nature of the relationships, the statistical evidence supports the validity of the observed associations between the variables.

4 References

1. Pang-Ning Tan, Steinbach, M., Vipin Kumar. (2014). Pearson new international edition : introduction to data mining. Pearson Addison Wesley.